STIC Search Report
EIC 2100

STIC Database Tracking Number: 169068

TO: Jason Proctor
Location: RND-5C31

Art Unit: 2123
Wednesday, June 28, 2006
Case Serial Number: 10/014,831

From: Lance Sealey
Location: EIC 2100
RND-4B11
Phone: 571-272-8666

Lance.Sealey@uspto.gov

Search Notes

Dear Jason,

Attached is the closest reference I could find.

Please feel free to call, email or visit me if you have any questions or concerns.

Lance

STIC EIC 2100 Search Request Form

193090

61

Today's Date: 6/15/2006

What date would you like to use to limit the search?

Priority Date: 12/15/2000 Other:

Name JASON PROCTOR

AU 2123 Examiner # 80434

Room # 5C31 Phone 23713

Serial # 10/014831

Format for Search Results (Circle One):

(PAPER) DISK EMAIL

Where have you searched so far?

(USP) DWPI (EPO) (JPO) (ACM) IBM TDB

(IEEE) INSPEC SPI Other internet

Is this a "Fast & Focused" Search Request? (Circle One) YES (NO)

A "Fast & Focused" Search is completed in 2-3 hours (maximum). The search must be on a very specific topic and meet certain criteria. The criteria are posted in EIC2100 and on the EIC2100 NPL Web Page at http://ptoweb/patents/stic/stic-tc2100.htm.

What is the topic, novelty, motivation, utility, or other specific details defining the desired focus of this search? Please include the concepts, synonyms, keywords, acronyms, definitions, strategies, and anything else that helps to describe the topic. Please attach a copy of the abstract, background, brief summary, pertinent claims and any citations of relevant art you have found.

Claim 1, step b. SEARCH PREP TIME: 60 MIN. - TERMINAL TIME 300 MIN.

- examining/analyzing software code prior to execution
  - identifies a loop (for, while, do, etc.) that might be infinite loop (possibly non-terminating)
    (This is as broad as it sounds - all loops could be "possibly non-terminating")
  - places a break, exit, or "loop termination" statement in the loop (which ensures that it will not be an infinite loop.)

→ may find good art in Software Engineering - testing code, "line coverage" "statement coverage" "block coverage" etc.

Thanks for anything you find!

STIC Searcher KANCE SEALEY Phone 2-8666

Date picked up 6/23/06 Date Completed 6/26/06

RECEIVED
JUN 15 2006

sira Search and Information Resources Administration

BY:.....

ABSTRACT OF THE DISCLOSURE

A method is provided for co-simulating a digital circuit using a simulation engine (45) which communicates with one or more first programming languages by means of a foreign language interface and which communicates directly with one or more second programming language. At least one first model (2, 3) or at least one first part of the digital circuit is provided in at least one high-level hardware description language which supports concurrent processes communicating with each other. The at least one first model is converted (50, 51) to at least one software model in the at least one first language. At least one second model (4, 5, 6) of at least one second part of the digital circuit is provided in the at least one second language.

00R00416

METHOD OF CO-SIMULATING A DIGITAL CIRCUIT

BACKGROUND OF THE INVENTION

1. FIELD OF THE INVENTION:

The present invention relates to a method of co-simulating a digital circuit. Such a method may be used as part of the design and manufacturing process of integrated circuits, for example of VLSI type.

2. DESCRIPTION OF THE RELATED ART:

Non-trivial digital hardware circuits are usually designed using a synthesis-based approach where the circuit is described in a Hardware Description Language (HDL) and then synthesised into hardware using a synthesis tool. VHDL (for example as disclosed in IEEE Computer Society, "IEEE Standard VHDL Language Reference Manual" New York. USA, March 1988. IEEE Std 1076-1987 and IEEE Computer Society, "IEEE Standard VHDL Language Reference Manual" New York, USA, June 1994. IEEE Std 1076-1993) and Verilog HDL (for example as disclosed in IEEE computer Society, "IEEE Standard Hardware Description Language Based on the Verilog Hardware Description Language." New York, USA 1996. IEEE Std 1364-1995) are commonly used hardware description languages. However, as circuit complexity continues to increase, there is a trend to use

higher-level hardware description languages, usually based on programming languages such as C (for example as disclosed in, Brian W. Kernighan and Dennis M. Ritchie, "The C Programming Language. Prentice-Hall, USA, second edition, 1988") and C++ (for example as disclosed in Bjarne Stroustrup, "The C++ programming language." Addison-Wesley series in computer science, Addison-Wesley, Reading, MA, USA) instead of register transfers.

Such languages allow the design of hardware in terms of algorithms. High-level synthesis tools (for example as disclosed in Daniel Gajski, Nikil Dutt, Allen Wu, and Steve Lin, "High-Level Synthesis, introduction to Chip and System Design." Kluwer Academic Publishers, Boston/Dordrecht/London, 1992) are then used to generate lower level HDL descriptions from the given algorithm-level descriptions. Similarly to software design, the use of a high-level language usually results in shorter design times.

In some systems, the high-level HDL used is simply a well known programming language. For instance System C (for example as disclosed in Synopsys Inc. "Overview of the

Open System C initiative," datasheet available on the internet from www.systemc.org,1999) uses C++ as a system description language. In other cases, a programming language with extensions relevant to hardware design is used. Examples of such systems include the Tangram system (as disclosed in K. van Berkel, J. Kessel, M. Roncken, R. Saeijs, and F. Schalij, "The VLSI-Programming Language Tangram and its Translation into Handshake Circuits", Proceeding of the European Design Automation Conference (EDAC 91), pages 384-389, Amsterdam, February 1991, IEEE, IEEE Computer Society Press and Kees van Berkel, "Handshake Circuits", volume 5 of Cambridge International Series on Parallel Computation, Cambridge University Press, Cambridge, UK, 1993) and the Bach system (as disclosed in Akihisa Yamada, Koichi Nishida, Ryoji Sakurai, Andrew Kay, Toshio Nomura and Takashi Kambe, ""Hardware synthesis with the BACH system"" International Symposium on Circuits and Systems, 1999 and in GB 231724S).

The language used by the Bach hardware compiler extends the C language with (amongst other features) constructs for expressing explicit parallelism and synchronous communication. The Bach language is based on the

Communicating Sequential Processes (CSP) model, which is disclosed in C.A.R. Hoare, "Communicating sequential processes." Communications of the ACM, 21 (8):666-677, August 1978 and C.A.R. Hoare, "Communicating Sequential Processes." Prentice-Hall International, Englewood Cliffs (NJ), USA, 1990, first edition published in 1985 by Prentice-Hall and which is a model of computation which supports concurrency. The Tangram language is also based on CSP.

Another important advantage of using a high-level HDL is faster simulation speeds due to the level of abstraction of the design description. Very fast simulation speeds can also be achieved by compilation based simulation (see for example L.T. Wang, N.E. Hoover, E.H. Porter, and J.J. Zasio. "SSIM: A software levelised compiled-code simulator", Proceeding of the 24th Design Automation Conference, pages 2-8, IEEE,IEEE Computer Society Press, 1987) where the hardware description is compiled into an executable format rather than interpreted by the simulation engine. In the case of using a sequential programming language (such as C++) as a HDL, a hardware description can be compiled and simulated simply by using a standard compiler for the particular language. If the

programming language used is extended with hardware design relevant features such as parallelism, then a hardware description can be converted into a sequential program before being compiled. For example, in the Tangram system, a hardware description can be converted into a C program as disclosed in Kees van Berkel, "Handshake Circuits," volume 5 of Cambridge University Press, Cambridge, UK, 1993. Also, JP 1121939 describes a simple mechanism for converting CSP features into a sequential language.

In systems comprising of one or more components, every component may be described in a language chosen for its particular strengths and expressiveness. For example, a hardware description language is used for hardware components and a software programming language is used for software components. It is therefore very common that the components in a system are described in several languages. Figure 4 of the accompanying drawings shows an example of such a system description. The complete system description comprises a plurality of component model descriptions which communicate with each other. The Bach C Language mentioned hereinbefore is used at 2 and 3 to provide descriptions of a demodulator component

and an error correction decoder component. The VHDL language mentioned hereinbefore is used at 4 and 5 to describe a RAM (random access memory) component and a Fast Fourier Transform (FFT) component. The C Language is used at 6 to describe a test bench component.

Since the verification of such a system is an essential part of its design process, it is required that the verification, or simulation, is fast as a lot of simulation data may have to be processed. This simulation process is often referred to as co-simulation because of the heterogeneous nature of the system. The different system component models need to communicate with each other during co-simulation, and known methods, such as that disclosed in US 5335191, can be used. One method for the co-simulation of a hardware component designed in a high-level HDL is to synthesise the hardware description into a lower-level HDL using a high-level synthesis tool or simulation engine 8 as illustrated in Figure 2 of the accompanying drawings, and then co-simulate the low-level description using the hardware simulation tool. However, this method does not take advantage of the fact that a high-level description can be used for simulation, and has the following

disadvantages:

(a) synthesis time overhead;

(b) slower simulation due to the use of a lower-level HDL;

(c) applicable only to synthesisable descriptions.

Hardware simulators presently available allow the simulation of models described in different HDLs, as well as foreign models, that is, models described using means other than the HDLs understood by the simulator. For instance, the latest standard of VHDL (for example as disclosed in IEEE Computer Society, "IEEE Standard VHDL Language Reference Manual," New York. USA, June 1994. IEEE Std 1076-1993) allows the specification of foreign entities. The Synopsys VSS simulator (as disclosed in Synopsys Inc. VSS Reference Manual. USA, 1998) provides a C Language Interface (CLI) (as disclosed in Synopsys Inc. VSS Interfaces Manual. USA, 1998) for the implementation of foreign entities using the C language. Similarly the Model Technology ModelSim simulator (as disclosed in Model Technology Inc. ModelSim SE/EE User's Manual. USA.1999) provides a Foreign Language Interface (FLI) for the same reason. The simulation engine described

in David A. Burgoon. A mixed-language simulator for concurrent engineering. In The Proceedings for the 1998 International Verilog HDL Conference and VHDL International Users Forum, US, March 1998. IEEE Computer Society is also capable of co-simulating C models with lower level Verilog Models. These particular methods apply only when the high-level hardware is described in a sequential language such as C. Further, the C code must be written in a special stimulus-response fashion, which is not purely algorithmic.

SUMMARY OF THE INVENTION

According to a first aspect of the invention, there is provided a method of co-simulating a digital circuit using a simulation engine which communicates with at least one first programming language by means of a foreign language interface and which communicates directly with at least one second programming or hardware description language, comprising the steps of:

(a) providing at least one first model of at least one first part of the digital circuit in at least one high-level hardware description language which supports

concurrent processes communicating with each other;

(b) converting the at least one first model to at least one software model in the at least one first language;

(c) providing at least one second model of at least one second part of the digital circuit in the at least one second language; and

(d) applying the at least one software model in the at least one first language and the at least one second model in the at least one second language to the simulation engine.

A high-level or behavioural hardware description is a description of a hardware component which specifies only the behaviour of the component and does not specify its physical architecture, such as the logical, arithmetic and storage components of which it is constituted, the clock rate and its timing. The behaviour is usually given as an algorithm. A high-level or behavioural hardware description language is a language in which the high-level description of the hardware can be described. High-level or behavioural hardware synthesis or

compilation is the process of generating a hardware low-level description from a high-level hardware description. Given a clock rate, this process infers the required logical, arithmetic and storage components, the signals connecting them, their timing and controlling logic.

The converting step (b) may comprise compiling the at least one first model in the at least one high-level hardware description language to the at least one software model in the at least one first language.

The at least one high-level hardware description language may be based on a communicating sequential processes model.

The at least one first part of the digital circuit may be represented in the at least one high-level hardware description language as a plurality of concurrent processes which communicate with each other and the converting step (b) may comprise converting the concurrent processes to a sequential software process. The software process may comprise at least one stimulus unit for detecting a predetermined stimulus and at least

one response unit for providing a predetermined response in response to the at least one stimulus unit. At least one of the response units may comprise a process response unit for performing a desired behaviour of the at least one first part of the digital circuit. The desired behaviour may comprise a plurality of discrete processes triggered by a common event and the process response unit may comprise a scheduler for scheduling the discrete processes and a process handler for performing the discrete processes in accordance with the scheduling. The scheduler may: form a list of active unhandled processes having respective exit points; choose from the list a current process; and select an entry point for the current process.

At each exit point, the scheduler may choose from the list a further current process and may select a further entry point for the further current process.

The converting step (b) may comprise: generating, for at least one of the discrete processes, software code including a program loop having a jump instruction and a loop termination condition; analysing the loop termination condition to determine whether it is possibly

non-terminating; and, if so replacing the jump instruction with an exit point.

At the exit point, the scheduler may place the at least one discrete process in the list of active unhandled processes with a new entry point.

According to a second aspect of the invention, there is provided a method of designing a digital circuit, comprising performing a method according to the first aspect of the invention, checking whether the result of the co-simulation is correct, checking whether the digital circuit is synthesisable, and generating a low-level hardware description of the digital circuit.

According to a third aspect of the invention, there is provided a method of manufacturing a digital circuit, comprising performing the method according to the second aspect of the invention and forming, from the low-level hardware description, an integrated circuit including the digital circuit.

According to a fourth aspect of the invention, there is provided an integrated circuit made by a method according

to the third aspect of the invention.

According to a fifth aspect of the invention, there is provided an apparatus for performing a method according to the first or second aspect of the invention.

The apparatus may comprise a computer programmed by a computer program.

According to a sixth aspect of the invention, there is provided a computer program for an apparatus according to the fifth aspect of the invention.

According to a seventh aspect of the invention, there is provided a storage medium containing a program according to the sixth aspect of the invention.

In the present method, a high-level sequential description of a synchronous hardware model can be generated automatically from a high-level hardware description based on a concurrent model of computation. The model description can be compiled into executable code and can be co-simulated with other system components. We therefore achieve the advantages of both high-level HDL

simulation and compilation based simulation for the co-simulation of a high-level hardware description with other system components.

An example of the use of this method is to co-simulate a hardware circuit described as an algorithm in a CSP-based high-level language with other system components during system design and development. For example, this method can be used for the fast co-simulation of hardware circuits described in the Bach C language with other system components. Figure 1 of the accompanying drawings illustrates the Bach hardware design flow, where hardware is described in the Bach high-level language, and a low-level synthesisable hardware description is generated automatically. Since the hardware designer uses a high-level language instead of a lower level one (such as VHDL), the design process is much quicker and therefore cheaper than traditional hardware design processes. The use of the present method allows the co-simulation of the hardware component to be done at the algorithm-level and is therefore much faster than a co-simulation process where lower-level hardware descriptions are used. As a result, the time spent in hardware design is reduced.

The advantages of this method include:

(a) allowing the co-simulation of the hardware component to be done at the algorithm-level and therefore:

(i) the simulation is independent of the target architecture;

(ii) the simulation is much faster than simulations at lower levels since the amount of detail that is simulated is much less.

(b) The component model can be compiled into the native code of the machine used for simulation from an algorithmic description. This approach offers very high simulation speeds.

(c) The generation of the hardware component model used for simulation does not require complex pre-computations and is therefore quite efficient.

(d) It is often desirable to co-simulate a hardware description with other system components during the early

stages of hardware development, that is, before an efficient and fully synthesisable hardware description has been developed. Since the hardware description used for co-simulation does not need to be synthesised into lower level descriptions, this co-simulation method can be used during these early stages of the design flow.

The above factors all contribute to the advantages associated with high-level hardware (and system) design flow:

(A) quicker time-to-market,

(B) and the ability to explore a large design space and hence develop more efficient designs.

The invention will be further described, by way of example, with reference to the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates the design flow of hardware using a high-level language and including a method of constituting an embodiment of the invention;

Serial No. 10/014,831

Amendments to the Claims

1. (currently amended) A method of co-simulating a digital circuit using a simulation engine which communicates with at least one first programming language by means of a foreign language interface and which communicates directly with at least one second programming or hardware description language, comprising the steps of:

(a) providing at least one first model of at least one first part of the digital circuit in at least one high-level hardware description language which supports concurrent processes communicating with each other;

(b) converting the at least one first model to at least one software model in the at least one first programming language, wherein converting includes generating, for at least one discrete process, software code including a program loop having a jump instruction and a loop termination condition, and analyzing the loop termination condition to determine whether it is ~~possibly non-terminating~~ not automatically determined to be definitely terminating and, if so, replacing the jump instruction with an exit point;

(c) providing at least one second model of at least one second part of the digital circuit in the at least one second language; and

(d) applying the at least one software model in the at least one first language and the at least one second model in the at least one second language to the simulation engine, so as to perform a simulation in order to obtain a simulation result.

2. (original) A method as claimed in claim 1, in which the converting step (b) comprises compiling the at least one first model in the at least one high-level hardware description language to the at least one software model in the at least one first language.

3. (original) A method as claimed in claim 1, in which the at least one high-level hardware description language is based on a communicating sequential processes model.

Next Contents Index

weak §103 motivation →

Liquid Common Lisp

The Loop Facility

The Loop Facility is an extensible iteration mechanism that provides you with a variety of ways to iterate and to accumulate values in a loop. This book discusses in detail the Loop Facility and its use, including the defined loop keywords and various error conditions.

---

this is narrower than required (inf loop vs. possibly non-terminating) and merely suggests a solution, no disclosure of an automatic system that actually places an exit in the loop

3 Error Conditions

3.1 About error conditions

Since `loop` is a macro, it translates the form you supply into another form. Thus, any syntactic error messages occur during macro expansion rather than during evaluation or compilation. When the Loop Facility parses the given form, it expects clauses to have loop keywords and data supplied in specific orders. Unexpected symbols or data generate an error and an error message. The message explains what the Loop Facility did not understand and where the error occurred.

There are some common errors that occur when the Loop Facility is used, and many are easily corrected. This chapter discusses these errors, their causes, and possible solutions. Examples that would generate errors are also supplied.

Problem: The loop form is being translated into an infinite loop; it never returns.

Reason: You forgot to specify a loop termination clause, or you supplied a termination clause that is not being executed.

Example: Lacking proper termination conditions.

```
> (loop)                                ; This is an infinite loop.
> (loop for i from 1 count i)           ; This is an infinite loop.
> (loop for i = 1 then 2                ; This is a third infinite loop.
        while i
        collect i)
```

Solution: To correct this problem, either specify a termination clause or revise the iteration conditions so that termination is possible.

Problem: An unrecognized form is encountered where a loop keyword is expected.

Reason: The Loop Facility parses the form that you give it token by token. Each token is one of the items in the form. If the Loop Facility cannot recognize a token as a loop keyword when a loop keyword is expected, it generates an error and a message.

The most common reasons that a token cannot be recognized as a loop keyword are as follows:

- the loop keyword is misspelled

```
> (loop untl (some-condition)           ; UNTIL is misspelled.
        do (print 'hi))
>>Error: Loop macro: Unrecognized form encountered where
loop-keyword expected: (... UNTL (SOME-CONDITION) DO (PRINT (QUOTE HI)) )


> (loop for i from 1 to 10
        od (print 'hi))                 ; DO is misspelled.
>>Error: Loop macro: Unrecognized form encountered where
loop-keyword expected: (... OD (PRINT (QUOTE HI)) )
```

Somewhat relevant, §4.5 suggests the claimed limitation could be novel in certain context

A Survey on Automatic Test Data Generation*

Jon Edvardsson,
Dept. of Computer and Information Science,
Linköping University, Sweden

E-mail: joned@ida.liu.se

Abstract

In order to reduce the high cost of manual software testing and at the same time to increase the reliability of the testing processes researchers and practitioners have tried to automate it. One of the most important components in a testing environment is an automatic test data generator — a system that automatically generates test data for a given program. Through the years several attempts in automatic test data generations have been made. The focus of this article is program-based generation, where the generation starts from the actual programs. Thus, techniques such as GUI-based and syntax-based test data generation are not an issue in this article.

In this article I present a survey on automatic test data generation techniques that can be found in current literature. Basic concepts and notions of test data generation as well as how a test data generator system works are described. Problems of automatic generation are identified and explained. Finally important and challenging future research topics are presented.

1. Introduction

Software testing accounts for 50% of the total cost of software development [1]. This cost could be reduced if the process of testing is automated. One way to do this would be to generate input data to the program to be tested — program-based test data generation.

Through the years a number of different methods for generating test data have been presented. In 1996 Ferguson and Korel [5] divided these methods in three classes: *random*, *path-oriented*, and *goal-oriented* test data generation. This is the most appropriate classification in terms of test data generation, although the problem of path selection is not considered separately. The selection of a path can largely affect the whole process of test data generation.

Figure 1 models a typical test data generator system, which consists of three parts: program analyzer, path selector and test data generator. The source code is run through a program analyzer, which produces the necessary data used by the path selector and the test data generator. The selector inspects the program data in order to find *suitable* paths. Suitable can for instance mean paths leading to a high code coverage. The paths are then given as argument to the test data generator which derives input values that exercise the given paths. The generator may provide the selector with feedback such as information concerning infeasible paths.

The structure of this paper is as follows. In section 2 basic concepts and notions are explained. Section 3 discusses the test data generator system with focus on the generator and the path selector. The program analyzer is not further investigated in this article. In section 4 the problems I have identified in test data generation are discussed. Finally, in section 5 conclusions are made and future research topics are presented.

2. Basic Concepts

A program $\mathcal{P}$ could be considered as a function, $\mathcal{P} : S \rightarrow R$, where S is the set of all possible inputs and R the set of all possible outputs. More formally S is the set of all vectors $\mathbf{x} = (d_1, d_2, \cdots, d_n)$ such that $d_i \in D_{x_i}$ where D_{x_i} is the domain of input variable x_i.

An input variable x of $\mathcal{P}$ is a variable that either appears as an input parameter of $\mathcal{P}$ or in an input statement of $\mathcal{P}$, e.g. `read(x)`. Execution of $\mathcal{P}$ for a certain input $\mathbf{x}$ is denoted by $\mathcal{P}(\mathbf{x})$.

A *control flowgraph*, or just flowgraph when context is clear, is a graphical representation of a pro-

*Published as: J. Edvardsson. A survey on automatic test data generation. In *Proceedings of the Second Conference on Computer Science and Engineering in Linköping*, pages 21–28. ECSEL, October 1999.

Serial No. 10/014,831

Amendments to the Claims

1. (currently amended) A method of co-simulating a digital circuit using a simulation engine which communicates with at least one first programming language by means of a foreign language interface and which communicates directly with at least one second programming or hardware description language, comprising the steps of:

(a) providing at least one first model of at least one first part of the digital circuit in at least one high-level hardware description language which supports concurrent processes communicating with each other;

(b) converting the at least one first model to at least one software model in the at least one first programming language, wherein converting includes generating, for at least one discrete process, software code including a program loop having a jump instruction and a loop termination condition, and analyzing the loop termination condition to determine whether it is ~~possibly non-terminating~~ not automatically determined to be definitely terminating and, if so, replacing the jump instruction with an exit point;

(c) providing at least one second model of at least one second part of the digital circuit in the at least one second language; and

(d) applying the at least one software model in the at least one first language and the at least one second model in the at least one second language to the simulation engine, so as to perform a simulation in order to obtain a simulation result.

2. (original) A method as claimed in claim 1, in which the converting step (b) comprises compiling the at least one first model in the at least one high-level hardware description language to the at least one software model in the at least one first language.

3. (original) A method as claimed in claim 1, in which the at least one high-level hardware description language is based on a communicating sequential processes model.

USPTO/ASRC Aerospace

EIC Reference Interview Form

SEARCHER: LANCE SEALEY

SERIAL #: 10/014,831

ACCESS #: 193090

INTERVIEW DATE: _____

OR

E-MAIL DATE: _____

(ATTACH E-MAIL)

☐ EXAMINER NOT AVAILABLE

☐ SRF SUFFICIENT

This form is used to provide supplementary information and clarify search requests.
Questions that are clearly answered on theSearch Request Form need not be repeated.
WRITE ADDITIONAL NOTES ON REVERSE.

QUESTION	Y if on SRF	NOTES
PRELIMINARY STRATEGY Appropriate? Too Broad/Narrow? Good Example from Examiner's Search Results?		CLAIM 1, STEP b. "POSSIBLY NON-TERMINATING" IS A LOOP THAT HAS NOT BEEN IDENTIFIED, BEFORE EXECUTION, AS ~~DEFINITELY TERMINATING~~ EITHER "DEFINITELY
NOVELTY Which concepts must be covered for a reference to be useful?		TERMINATING" OR "DEFINITELY AN ENDLESS LOOP"; e.g. FOR int i; i ≤ 8; i++
APPLICATIONS How will this invention be applied? On which (if any) subject area or application should search focus?		i = 50 - N END. SINCE THIS IS A DIFFICULT CONCEPT TO EXPRESS IN A
KEY TERMS Terms of Art/Acronyms/ Professional Jargon Synonyms Terms to avoid		SEARCH, I [redacted] WILL FOCUS ON THE IDEAS OF ANALYZING SOFTWARE CODE BEFORE EXECUTION OR PREVENTING ENDLESS LOOPS. I HOPING I WOULD FIND THE IDEA OF DETECTING "NON-TERMINATING" IN MY RESULTS. AS FOR THE SOFTWARE ENGINEERING TERMS SUGGESTED
DATABASES Foreign Patents Internet Search (recommended search engines or websites)		BY THE EXAMINER, AFTER LOOKING THEM UP AND READING THE CLAIMS AND THE SPECIFICATION OF THE APPLICATION, I DETERMINED THAT THE TERMS DID NOT APPLY TO
RESULTS FORMAT Y N Tagged? Y N Highlighted? Y N Include Inventor Search (if no valuable results) ?		WHAT I WAS ASKED TO SEARCH, AND I THEREFORE DID NOT USE THEM. (NOTE THAT THE EXAMINER'S STATEMENT WAS A HELPFUL SUGGESTION, NOT A COMMAND.)
DATE What date would you like to use to limit the search?		**Priority Date:** _____ **Other Date:** _____

Search Chronology

Date Searcher Picked Up: _____

Date Completed: _____

Searcher Prep & Review Time: _____

Online Time: _____

Clerical Prep Time: _____

Type of Search

NA Sequence (#) _____

AA Sequence (#) _____

Structure (#) _____

Text _____

Litigation _____

Patent Family _____

Other _____

Vendors and cost where applicable

STN _____

Dialog _____

Questel/Orbit _____

Lexis/Nexis _____

Sequence Systems _____

WWW/Internet _____

Other (specify) _____

Published before December 1999
Terms used **update endless loop**

Found **62** of **106,921**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 61 - 62 of 62 Result page: previous 1 2 3 **4**

Relevance scale

61 A hierarchical structure for fault tolerant reactive programs

Andrea Clematis, Vittoria Gianuzzi

March 1993 **Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing: states of the art and practice**

Publisher: ACM Press

Full text available: pdf(613.91 KB) Additional Information: full citation, references, index terms

Keywords: backward error recovery, concurrent programming, software fault tolerance, transaction based systems

62 Research issues in spatial databases

O. Guenther, A. Buchmann

December 1990 **ACM SIGMOD Record**, Volume 19 Issue 4

Publisher: ACM Press

Full text available: pdf(753.81 KB) Additional Information: full citation, citings, index terms

Results 61 - 62 of 62 Result page: previous 1 2 3 **4**

```
Set      Items   Description
S1       46453   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(10N)(-
             CODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBR-
             OUTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
              ? OR ITERATION? ?)
S2          36   S1(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
             OR PRE()COMPIL????? OR PREEXECUT???)
S3        1993   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
             RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
             ?)
S4           0   S2 AND S3
S5     3759665   STOP???? OR PREVENT????? OR CURTAIL??? OR IMPED??? OR INTE-
             RRUPT? OR DISABL??? OR DEACTIVAT??? OR DE()ACTIVAT???
S6     1695744   BREAK??? OR EXIT??? OR TERMIN??? OR BRANCH??? OR (CONTROL(-
             2W)FLOW)
S7     6096217   UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTER-
             ??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR -
             INSERT??? OR ADD???
S8          20   (S1 OR S5) AND S3 AND S6 AND S7
S9          18   S8 NOT (AD=(19991215:20021215) OR AD=(20021216:20051215) OR
              AD=(20051216:20060628))
S10          4   AU=((ZAMMIT V? OR ZAMMIT, V?) AND (KAY A? OR KAY, A?))
S11         11   (S3 AND (IC=(G06F-009/44 OR G06F-017/50) OR MC=T01)) NOT (-
             S9 OR S10)
? show files
File 347:JAPIO Dec 1976-2005/Dec(Updated 060404)
         (c) 2006 JPO & JAPIO
File 350:Derwent WPIX 1963-2006/UD,UM &UP=200640
         (c) 2006 The Thomson Corp.
?
```

10/5/3 (Item 1 from file: 350)
DIALOG(R)File 350:Derwent WPIX

014747738 **Image available**
WPI Acc No: 2002-568442/200261
XRPX Acc No: N02-450042

Digital circuit co-simulation method for integrated circuit designing involves converting models of digital circuit into software models and applying the models to simulation engine depending on description languages

Patent Assignee: SHARP KK (SHAF); KAY A (KAYA-I); ZAMMIT V (ZAMM-I)
Inventor: **KAY A ; ZAMMIT V**

Number of Countries: 003 Number of Patents: 003
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
GB 2370134	A	20020619	GB 200030735	A	20001215	200261 B
US 20020083420	A1	20020627	US 200114831	A	20011211	200261
JP 2002183234	A	20020628	JP 2001378776	A	20011212	200261

Priority Applications (No Type Date): GB 200030735 A 20001215
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
GB 2370134	A		56	G06F-017/50	
US 20020083420	A1			G06F-009/44	
JP 2002183234	A		23	G06F-017/50	

Abstract (Basic): GB 2370134 A

NOVELTY - A portion of digital circuit is defined by primary model in high level hardware description language and is converted into software model. The next portion of digital circuit is defined by a secondary model using different language. The models of digital circuit produced corresponding to two languages are applied to a simulation engine (45).

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

(1) Method of designing digital circuit;
(2) Method of manufacturing digital circuit;
(3) Integrated circuit;
(4) Apparatus for designing digital circuit;
(5) Program for designing digital circuit; and
(6) Storage medium having program for designing digital circuit.

USE - For designing of IC e.g. very large scale integrated circuit.

ADVANTAGE - High speed simulation of digital circuit is achieved, as the complex precomputations in the generation of hardware component model is reduced.

DESCRIPTION OF DRAWING(S) - The figure shows the overall method used for generating simulation model for digital circuit.

Simulation engine (45)

pp; 56 DwgNo 10/30

Title Terms: DIGITAL; CIRCUIT; CO; SIMULATE; METHOD; INTEGRATE; CIRCUIT; DESIGN; CONVERT; MODEL; DIGITAL; CIRCUIT; SOFTWARE; MODEL; APPLY; MODEL; SIMULATE; ENGINE; DEPEND; DESCRIBE; LANGUAGE
Derwent Class: T01
International Patent Class (Main): G06F-009/44; G06F-017/50
File Segment: EPI

9/5/8 (Item 8 from file: 347)
DIALOG(R)File 347:JAPIO

01586845 **Image available**
PROGRAM RUNAWAY **PREVENTING** METHOD

PUB. NO.: 60-065345 [JP 60065345 A]
PUBLISHED: April 15, 1985 (19850415)
INVENTOR(s): NISHIURA YOSHIKAZU
APPLICANT(s): SHARP CORP [000504] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 58-174988 [JP 83174988]
FILED: September 20, 1983 (19830920)
INTL CLASS: [4] G06F-011/00; G05B-009/02
JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units); 22.3 (MACHINERY -- Control & Regulation)
JAPIO KEYWORD:R097 (ELECTRONIC MATERIALS -- Metal Oxide Semiconductors, MOS); R129 (ELECTRONIC MATERIALS -- Super High Density Integrated Circuits, LSI & GS
JOURNAL: Section: P, Section No. 381, Vol. 09, No. 203, Pg. 16, August 21,. 1985 (19850821)

ABSTRACT

PURPOSE: To **prevent** the runaway of a program by **modifying** the contents of a program counter with the contents of another register and setting forcibly the contents of another register to the prescribed value.

CONSTITUTION: A system has an instruction to load the contents of an accumulator R (4 bits) to the lower 4 bits of a program counter PC (8 bits). When the contents of the PC go to 1FHEX, for example, according to the progress of a program, a program **branching** instruction ATPL is executed. When this execution is closed, the contents of the R are once set forcibly at ''0''. In this case, however, the restriction is previously set so that the lower 4 bits of the PC are not set at ''0''. The contents of the PC and the R always go to 1FHEX and 0HEX respectively after the execution of instruction ATPL even in an unexpected case where the contents of the PC and the R go to 1FHEX and FHEX respectively. Then the contents of the PC are **branched** to 10HEX in the next cycle. Thus an **endless loop** is **prevented**.
?

```
Set     Items   Description
S1     396399   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(3N)(C-
            ODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBRO-
            UTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
            ? OR ITERATION? ?)
S2        64   S1(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
            OR PRE()COMPIL????? OR PREEXECUT???)
S3     851402   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(10N)(-
            CODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBR-
            OUTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
             ? OR ITERATION? ?)
S4       145   S3(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
            OR PRE()COMPIL????? OR PREEXECUT???)
S5      4522   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
            RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
            ?)
S6         0   S4(10N)S5
S7       180   (UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTE-
            R??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR
            INSERT???)(10N)S5
S8         4   S7(10N)(BREAK??? OR EXIT??? OR TERMIN???)
? show files
File 275:Gale Group Computer DB(TM) 1983-2006/Jun 23
         (c) 2006 The Gale Group
File  47:Gale Group Magazine DB(TM) 1959-2006/Jun 26
         (c) 2006 The Gale group
File  16:Gale Group PROMT(R) 1990-2006/Jun 23
         (c) 2006 The Gale Group
File 624:McGraw-Hill Publications 1985-2006/Jun 23
         (c) 2006 McGraw-Hill Co. Inc
File 484:Periodical Abs Plustext 1986-2006/Jun W3
         (c) 2006 ProQuest
File 613:PR Newswire 1999-2006/Jun 24
         (c) 2006 PR Newswire Association Inc
File 813:PR Newswire 1987-1999/Apr 30
         (c) 1999 PR Newswire Association Inc
File 239:Mathsci 1940-2006/Aug
         (c) 2006 American Mathematical Society
File 370:Science 1996-1999/Jul W3
         (c) 1999 AAAS
File 696:DIALOG Telecom. Newsletters 1995-2006/Jun 23
         (c) 2006 Dialog
File 621:Gale Group New Prod.Annou.(R) 1985-2006/Jun 23
         (c) 2006 The Gale Group
File 674:Computer News Fulltext 1989-2006/Jun W2
         (c) 2006 IDG Communications
File  88:Gale Group Business A.R.T.S. 1976-2006/Jun 16
         (c) 2006 The Gale Group
File 369:New Scientist 1994-2006/Jun W2
         (c) 2006 Reed Business Information Ltd.
File 160:Gale Group PROMT(R) 1972-1989
         (c) 1999 The Gale Group
File 635:Business Dateline(R) 1985-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File  15:ABI/Inform(R) 1971-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File   9:Business & Industry(R) Jul/1994-2006/Jun 23
         (c) 2006  The Gale Group
File  13:BAMP 2006/Jun W3
         (c) 2006  The Gale Group
File 810:Business Wire 1986-1999/Feb 28
```

(c) 1999 Business Wire
File 610:Business Wire 1999-2006/Jun 24
(c) 2006 Business Wire.
File 647:CMP Computer Fulltext 1988-2006/Jul W4
(c) 2006 CMP Media, LLC
File 98:General Sci Abs 1984-2005/Jan
(c) 2006 The HW Wilson Co.
File 148:Gale Group Trade & Industry DB 1976-2006/Jun 26
(c)2006 The Gale Group
File 634:San Jose Mercury Jun 1985-2006/Jun 22
(c) 2006 San Jose Mercury News
File 256:TecInfoSource 82-2006/Aug
(c) 2006 Info.Sources Inc
File 636:Gale Group Newsletter DB(TM) 1987-2006/Jun 23
(c) 2006 The Gale Group
?

```
Set     Items   Description
S1       4522   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
            RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
            ?)
S2          0   AU=((ZAMMIT V? OR ZAMMIT, V?) AND (KAY A? OR KAY, A?))
S3          0   AU=(ZAMMIT V? OR ZAMMIT, V? OR KAY A? OR KAY, A?) AND S1 A-
            ND (PD<19991215 OR PY<2000)
S4        120   (STOP???? OR PREVENT????? OR CURTAIL??? OR IMPED??? OR INT-
            ERRUPT? OR DISABL??? OR DEACTIVAT??? OR DE()ACTIVAT???)(5N)S1
S5         72   RD  (unique items)
S6         56   S5 AND (PD<19991215 OR PY<2000)
? show files
File 275:Gale Group Computer DB(TM) 1983-2006/Jun 23
         (c) 2006 The Gale Group
File  47:Gale Group Magazine DB(TM) 1959-2006/Jun 26
         (c) 2006 The Gale group
File  16:Gale Group PROMT(R) 1990-2006/Jun 23
         (c) 2006 The Gale Group
File 624:McGraw-Hill Publications 1985-2006/Jun 23
         (c) 2006 McGraw-Hill Co. Inc
File 484:Periodical Abs Plustext 1986-2006/Jun W3
         (c) 2006 ProQuest
File 613:PR Newswire 1999-2006/Jun 24
         (c) 2006 PR Newswire Association Inc
File 813:PR Newswire 1987-1999/Apr 30
         (c) 1999 PR Newswire Association Inc
File 239:Mathsci 1940-2006/Aug
         (c) 2006 American Mathematical Society
File 370:Science 1996-1999/Jul W3
         (c) 1999 AAAS
File 696:DIALOG Telecom. Newsletters 1995-2006/Jun 23
         (c) 2006 Dialog
File 621:Gale Group New Prod.Annou.(R) 1985-2006/Jun 23
         (c) 2006 The Gale Group
File 674:Computer News Fulltext 1989-2006/Jun W2
         (c) 2006 IDG Communications
File  88:Gale Group Business A.R.T.S. 1976-2006/Jun 16
         (c) 2006 The Gale Group
File 369:New Scientist 1994-2006/Jun W2
         (c) 2006 Reed Business Information Ltd.
File 160:Gale Group PROMT(R) 1972-1989
         (c) 1999 The Gale Group
File 635:Business Dateline(R) 1985-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File  15:ABI/Inform(R) 1971-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File   9:Business & Industry(R) Jul/1994-2006/Jun 23
         (c) 2006  The Gale Group
File  13:BAMP 2006/Jun W3
         (c) 2006  The Gale Group
File 810:Business Wire 1986-1999/Feb 28
         (c) 1999 Business Wire
File 610:Business Wire 1999-2006/Jun 24
         (c) 2006 Business Wire.
File 647:CMP  Computer Fulltext 1988-2006/Jul W4
         (c) 2006 CMP Media, LLC
File  98:General Sci Abs 1984-2005/Jan
         (c) 2006 The HW Wilson Co.
File 148:Gale Group Trade & Industry DB 1976-2006/Jun 26
         (c)2006 The Gale Group
File 634:San Jose Mercury  Jun 1985-2006/Jun 22
```

FULL TEXT NPL/
NPL INVENTOR

(c) 2006 San Jose Mercury News
File 256:TecInfoSource 82-2006/Aug
(c) 2006 Info.Sources Inc
File 636:Gale Group Newsletter DB(TM) 1987-2006/Jun 23
(c) 2006 The Gale Group
?

```
Set     Items   Description
S1       4522   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
            RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
            ?)
S2        207   (UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTE-
            R??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR
            INSERT??? OR ADD???)(10N)S1
S3         12   S2(10N)(BREAK??? OR EXIT??? OR TERMIN??? OR BRANCH??? OR C-
            ONTROL(2W)FLOW)
? show files
File 275:Gale Group Computer DB(TM) 1983-2006/Jun 23
         (c) 2006 The Gale Group
File  47:Gale Group Magazine DB(TM) 1959-2006/Jun 26
         (c) 2006 The Gale group
File  16:Gale Group PROMT(R) 1990-2006/Jun 23
         (c) 2006 The Gale Group
File 624:McGraw-Hill Publications 1985-2006/Jun 23
         (c) 2006 McGraw-Hill Co. Inc
File 484:Periodical Abs Plustext 1986-2006/Jun W3
         (c) 2006 ProQuest
File 613:PR Newswire 1999-2006/Jun 25
         (c) 2006 PR Newswire Association Inc
File 813:PR Newswire 1987-1999/Apr 30
         (c) 1999 PR Newswire Association Inc
File 239:Mathsci 1940-2006/Aug
         (c) 2006 American Mathematical Society
File 370:Science 1996-1999/Jul W3
         (c) 1999 AAAS
File 696:DIALOG Telecom. Newsletters 1995-2006/Jun 23
         (c) 2006 Dialog
File 621:Gale Group New Prod.Annou.(R) 1985-2006/Jun 23
         (c) 2006 The Gale Group
File 674:Computer News Fulltext 1989-2006/Jun W2
         (c) 2006 IDG Communications
File  88:Gale Group Business A.R.T.S. 1976-2006/Jun 16
         (c) 2006 The Gale Group
File 369:New Scientist 1994-2006/Jun W3
         (c) 2006 Reed Business Information Ltd.
File 160:Gale Group PROMT(R) 1972-1989
         (c) 1999 The Gale Group
File 635:Business Dateline(R) 1985-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File  15:ABI/Inform(R) 1971-2006/Jun 24
         (c) 2006 ProQuest Info&Learning
File   9:Business & Industry(R) Jul/1994-2006/Jun 23
         (c) 2006  The Gale Group
File  13:BAMP 2006/Jun W3
         (c) 2006  The Gale Group
File 810:Business Wire 1986-1999/Feb 28
         (c) 1999 Business Wire
File 610:Business Wire 1999-2006/Jun 25
         (c) 2006 Business Wire.
File 647:CMP  Computer Fulltext 1988-2006/Jul W4
         (c) 2006 CMP Media, LLC
File  98:General Sci Abs 1984-2005/Jan
         (c) 2006 The HW Wilson Co.
File 148:Gale Group Trade & Industry DB 1976-2006/Jun 26
         (c)2006 The Gale Group
File 634:San Jose Mercury  Jun 1985-2006/Jun 22
         (c) 2006 San Jose Mercury News
File 256:TecInfoSource 82-2006/Aug
```

FULL TEXT NPC

(c) 2006 Info.Sources Inc
File 636:Gale Group Newsletter DB(TM) 1987-2006/Jun 23
(c) 2006 The Gale Group
?

```
Set      Items   Description
S1     1079868   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(3N)(C-
             ODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBRO-
             UTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
             ? OR ITERATION? ?)
S2          93   S1(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
             OR PRE()COMPIL????? OR PREEXECUT???)
S3     1138269   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(10N)(-
             CODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBR-
             OUTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
              ? OR ITERATION? ?)
S4         201   S3(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
             OR PRE()COMPIL????? OR PREEXECUT???)
S5        3561   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
             RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
             ?)
S6           0   S4(10N)S5
S7         279   (UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTE-
             R??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR
             INSERT???)(10N)S5
S8           6   S7(10N)(BREAK??? OR EXIT??? OR TERMIN???)
S9           1   AU=((ZAMMIT V? OR ZAMMIT, V?) AND (KAY A? OR KAY, A?))
S10          0   (AU=(ZAMMIT V? OR ZAMMIT, V? OR KAY A? OR KAY, A?) AND S5)
             NOT (PD=(19991215:20021215) OR PD=(20021215:20060623))
S11         60   S5 AND IC=(G06F-009/44)
S12         24   S11 NOT (PD=(19991215:20021215) OR PD=(20021215:20060623))
S13        237   ((STOP???? OR PREVENT????? OR CURTAIL??? OR IMPED??? OR IN-
             TERRUPT? OR DISABL??? OR DEACTIVAT??? OR DE()ACTIVAT???)(5N)S-
             5) NOT (S8:S9 OR S12)
S14          0   S13(100N)((UPDAT??? OR PLAC??? OR CHANG??? OR MODIF???????-
             ?? OR ALTER??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W-
             )DAT??? OR INSERT???)(10N)(BREAK??? OR EXIT??? OR TERMIN???))
S15         41   S13 AND ((UPDAT??? OR PLAC??? OR CHANG??? OR MODIF?????????
              OR ALTER??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)D-
             AT??? OR INSERT???)(10N)(BREAK??? OR EXIT??? OR TERMIN???))
S16         62   S13 NOT (S8:S9 OR S12 OR S15 OR PD=(19991215:20021215) OR -
             PD=(20021215:20060623))
? show files
File 348:EUROPEAN PATENTS 1978-2006/ 200625
         (c) 2006 European Patent Office
File 349:PCT FULLTEXT 1979-2006/UB=20060622,UT=20060615
         (c) 2006 WIPO/Univentio
?
```

```
Set     Items   Description
S1       3561   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
            RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
            ?)
S2        298   (UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTE-
            R??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR
            INSERT??? OR ADD???)(10N)S1
S3         10   S2(10N)(BREAK??? OR EXIT??? OR TERMIN??? OR BRANCH??? OR C-
            ONTROL(2W)FLOW)
? show files
File 348:EUROPEAN PATENTS 1978-2006/ 200625
         (c) 2006 European Patent Office
File 349:PCT FULLTEXT 1979-2006/UB=20060622,UT=20060615
         (c) 2006 WIPO/Univentio
```

FULL TEXT
PATENT

```
Set     Items   Description
S1    1046445   (ANALY???? OR EXAMIN????? OR EVALUAT??? OR CHECK???)(10N)(-
            CODE? ? OR INSTRUCTIONS OR PROCEDURE? ? OR ROUTINE? ? OR SUBR-
            OUTINE? ? OR MODULE? ? OR SCRIPT? ? OR APPLICATION? ? OR LOOP?
             ? OR ITERATION? ?)
S2        119   S1(10N)(((PRIOR OR BEFORE)(2W)EXECUT???) OR PRECOMPIL?????
            OR PRE()COMPIL????? OR PREEXECUT???)
S3       1475   (INFINITE OR ENDLESS OR NONTERMIN????? OR UNENDING OR ALDE-
            RSON OR NON()(TERMIN??? OR END???))(2W)(LOOP? ? OR ITERATION?
            ?)
S4          0   S2 AND S3
S5    2344617   STOP???? OR PREVENT????? OR CURTAIL??? OR IMPED??? OR INTE-
            RRUPT? OR DISABL??? OR DEACTIVAT??? OR DE()ACTIVAT???
S6    2200183   BREAK??? OR EXIT??? OR TERMIN??? OR BRANCH??? OR (CONTROL(-
            2W)FLOW)
S7   14146205   UPDAT??? OR PLAC??? OR CHANG??? OR MODIF????????? OR ALTER-
            ??? OR CONVERS??? OR CONVERT??? OR EDIT??? OR UP(W)DAT??? OR -
            INSERT??? OR ADD???
S8         19   (S1 OR S5) AND S3 AND S6 AND S7
S9         18   RD  (unique items)
? show files
File   2:INSPEC 1898-2006/Jun W3
         (c) 2006 Institution of Electrical Engineers
File   6:NTIS 1964-2006/Jun W3
         (c) 2006 NTIS, Intl Cpyrght All Rights Res
File   8:Ei Compendex(R) 1970-2006/Jun W3
         (c) 2006 Elsevier Eng.  Info. Inc.
File  34:SciSearch(R) Cited Ref Sci 1990-2006/Jun W4
         (c) 2006 Inst for Sci Info
File  35:Dissertation Abs Online 1861-2006/Jun
         (c) 2006 ProQuest Info&Learning
File  56:Computer and Information Systems Abstracts 1966-2006/Jun
         (c) 2006 CSA.
File  57:Electronics & Communications Abstracts 1966-2006/Jun
         (c) 2006 CSA.
File  60:ANTE: Abstracts in New Tech & Engineer 1966-2006/Jun
         (c) 2006 CSA.
File  65:Inside Conferences 1993-2006/Jun 28
         (c) 2006 BLDSC all rts. reserv.
File  94:JICST-EPlus 1985-2006/Mar W4
         (c)2006 Japan Science and Tech Corp(JST)
File  95:TEME-Technology & Management 1989-2006/Jul W1
         (c) 2006 FIZ TECHNIK
File  99:Wilson Appl. Sci & Tech Abs 1983-2006/May
         (c) 2006 The HW Wilson Co.
File 111:TGG Natl.Newspaper Index(SM) 1979-2006/Jun 19
         (c) 2006 The Gale Group
File 144:Pascal 1973-2006/Jun W1
         (c) 2006 INIST/CNRS
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec
         (c) 1998 Inst for Sci Info
File 636:Gale Group Newsletter DB(TM) 1987-2006/Jun 27
         (c) 2006 The Gale Group
?
```

Web Results 91 - 100 of about 8,110,000 for (endless OR infinite OR alderson) loop (update OR modify OR edit OR add)(break OR exit OR branch). (0.23 seconds)

MySQL Bugs: #121: ./configure goes into infinite loop on mit-pthreads
configure goes into infinite loop on mit-pthreads MySQL Part of configure reads: (cd mit-pthreads; sh ./configure) I invoked configure as so: sh -v . ...
bugs.mysql.com/bug.php?id=121 - 9k - Cached - Similar pages

REALbasic University: Column 107
We still have a infinite loop and must force quit our app. ... But just in case there's some other odd tag situation, let's add an emergency exit: ...
www.applelinks.com/rbu/107/ - 42k - Cached - Similar pages

GNU Emacs Lisp Reference Manual: Debugging
Once you have the debugger running in the middle of the infinite loop, you can proceed from ... The Debugger mode c and r commands exit the recursive edit; ...
www.cs.huji.ac.il/~osigor/emacs/elisp-help/elisp-manref/elisp_18.html - 110k - Cached - Similar pages

The Burning Edge » Gecko 1.8 branch
Gecko 1.8 branch checkins between 2005-11-07 12:00 and 2005-11-11 16:05 ... Fixed: 314684 - Endless update loop from firefox 1.5 beta2 to 1.5 rc1 if 1.0.x ...
www.squarefree.com/burningedge/categories/gecko-18-branch/ - 37k - Cached - Similar pages

<?php ...
1.6 - Fix for endless loop bug on certain special characters ... line-break would make more sense. You may edit these characters by ...
www.greywyvern.com/code/php/htmlwrap.phps - 27k - Cached - Similar pages

Troubleshooting - Document routing messages, EHLE039E - EHLE087E
Potential loop; check your connector criteria. Explanation:. A loop might occur because the connector does not contain proper continuation or branch ...
publib.boulder.ibm.com/infocenter/cmgmt/v8r3m0/topic/com.ibm.troubleshooting.doc/frnm2mst39.htm - 19k - Cached - Similar pages

Tutorial 1 - Basic SDL
This is our main, infinite loop. The if statements update our picture location ... break; case SDLK_UP: upPressed = TRUE; break; case SDLK_ESCAPE: exit(0); ...
pgdc.purdue.org/sdltutorial/01/ - 9k - Cached - Similar pages

CVS Update: xc (branch: trunk)
CVS Update: xc (branch: trunk). Thomas Hellstrom xorg-commit at cvs.freedesktop.org ... up in an endless loop if the offending region refuses to be removed. ...
lists.freedesktop.org/archives/xorg-commit/2005-May/002996.html - 3k - Cached - Similar pages

Timeline - gregarius - Trac
00:57 Ticket #420 (defect) closed by mdodoo: fixed: Turns out the infinite loop (and hence crash) was caused by a variable ... ...
svn.gregarius.net/trac/timeline - 21k - Cached - Similar pages

Recent changes that may break your gadgets - MicrosoftGadgets.com
We do have plans to add tabs to let you organize more content on your page, ... or Stack Overflow errors as the code ends up going into an infinite loop. ...
microsoftgadgets.com/forums/1438/ShowPost.aspx - 59k - Cached - Similar pages

◄ Gooooooooo o o o o o o o o o ogle ►
Result Page: Previous 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 Next

Google Scholar BETA (endless OR infinite OR alderson) loop (updat Search Advanced Scholar Search Scholar Preferences Scholar Help

Scholar Results 91 - 100 of about 21,200 for **(endless OR infinite OR alderson) loop (update OR edit OR modify OR add) (break OR exit)**. (0.28 seconds)

All articles Recent articles

A theory of discrete patterns and their implementation in SNOBOL4
JF Gimpel - Communications of the ACM, 1973 - portal.acm.org
... It is a simple matter to **modify** the algorithm to operate ... **BREAK**('C') is a transmitter and so is tAB' so ... null string and if s fails, an **infinite loop** would occur ...
Cited by 19 - Web Search

Challenges in Getting Formal with Viruses - group of 4 »
A Lakhotia, PK Singh - Virus Bulletin, 2003 - cacs.louisiana.edu
... For example, to avoid getting into an **infinite loop**, and thereby ... determine whether the two functions are in a **loop**. ... will not be too hard to **modify** a compiler ...
Cited by 11 - View as HTML - Web Search

Dimensional regularization of the path integral in curved space on an infinite time interval - group of 5 »
F Bastianelli, O Corradini, P van Nieuwenhuizen - Arxiv preprint hep-th/0007105, 2000 - arxiv.org
... to **modify** the counterterm V CT since such a ... parts of (2) already at the two-**loop** level. ... counterterm of dimensional regularization on the **infinite** time interval ...
Cited by 13 - View as HTML - Web Search

Affine relationships among variables of a program
MA Karr - Acta Informatica, 1976 - Springer
... We shall **modify** the matrix Al*-) to obtain A (1. The ... A (,-11 by flo and **add** it to row i of ... our current concern here that we have an **infinite loop**; rather, that ...
Cited by 113 - Web Search

Making Flash Animations and Incorporating Them in Web Pages - group of 2 »
RR Silbar - swcp.com
... right-click menu as usual), rename it, and then **edit** the four ... You're in an **endless loop**! ... that it does stop when the button appears – no more **infinite loop**. ...
View as HTML - Web Search

... crossover from perturbative to nonperturbative physics in QCD become a phase transition at infinite ... - group of 3 »
J Kiskis, RN FIU, H Neuberger - Arxiv preprint hep-lat/0308033, 2003 - arxiv.org
... At **infinite** N, six phases are encountered as b is var ... still is present, but the 1 × 1 **Wilson loop** now has ... the values of b we used, each SU(2) **update** was done ...
Cited by 26 - View as HTML - Web Search

TEAMS: Testability Engineering and Maintenance System - group of 2 »
KR Pattipati, V Raghavan, M Shakeri, S Deb, R ... - American Control Conference, 1994 - ieeexplore.ieee.org
... enables the user to graphically enter, **modify** and integrate ... algo- rithm from entering into an **infinite loop** when a ... Feedback **loop** analysis A system is said to ...
Cited by 8 - Web Search - BL Direct

Chain Reactions in Networks - group of 5 »
U Manber - Computer, 1990 - doi.ieeecs.org
... It presents several examples of actual **break**- downs, which ... caused a **loop** of routing **update** messages that ... usual vacation reply, and an **infinite loop** will result. ...
Cited by 7 - Web Search - Library Search

Animating equations on the Web - group of 4 »
RR Silbar - Computing in Science & Engineering [see also IEEE ..., 2000 - ieeexplore.ieee.org
... right-click menu as usual), rename it, and then **edit** the four ... You're in an **endless loop**! ... that it now stops when the button appears—no more **infinite loop**. ...
Cited by 2 - Web Search

Real-time avionics in Ada 83
DK Silvasi-Patchin - Proceedings of the conference on TRI-Ada'95: Ada's role in ..., 1995 - portal.acm.org
... procedure initiates all relevant timers and goes into an **infinite loop** (labeled as ... Idle **Loop** I ... **Update** TM-Bus Circular Buffer Pointers Periodic Built-In Test P-IT ...
Cited by 2 - Web Search

◄ Gooooooooo o o o o o o o o o ogle ►
Result Page: **Previous** 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 **Next**

(endless OR infinite OR alderson) l Search

Google Home - About Google - About Google Scholar

Google Scholar BETA (endless OR infinite OR alderson) loop (upc Search

Advanced Scholar Search
Scholar Preferences
Scholar Help

Scholar Results 91 - 100 of about 13,500 for **(endless OR infinite OR alderson) loop (update OR edit OR modify) (automatic OR automated)**. (0.19 seconds)

All articles Recent articles

and YP Gupta* Abstract Stabilizing properties based on monotonic behaviour of the solution of the ...
SD FINITE-HORIZON, IN BOUND - Control and Intelligent Systems, 2006 - actapress.com
... FINITE-HORIZON H∞CONTROL WITH GUARANTEED **INFINITE** NORM BOUND ... time from which the **controller gain update** is stopped ... wk − w* k, as the closed-**loop** sys- tem ...
Web Search

Power control and clustering in ad hoc networks - group of 20 »
V Kawadia, PR Kumar - INFOCOM 2003. Twenty-Second Annual Joint Conference of the ... - ieeexplore.ieee.org
... control in this fashion leads to **automatic** clustering in ... to the kernel routing table and to **modify** the forwarding ... back to node S, and we have an **infinite loop**. ...
Cited by 120 - Web Search - BL Direct

Asynchronous backtracking without adding links: a new member in the abt family - group of 4 »
C Bessiere, A Maestre, I Brito, P Meseguer - Artificial Intelligence, 2005 - lirmm.fr
... procedure **Update**(myAgentV iew, newAssig) 1 add(newAssig, myAgentV iew ... of our agents can fall into an **infinite loop**. ... Interestingly, it is possible to **modify** ABT ...
Cited by 11 - View as HTML - Web Search

String stability of interconnected systems: an application to platooning in **automated** highway ... - group of 3 »
D Swaroop - 1994 - path.berkeley.edu
... ical point of view. this study extends the concepts of stability to a countably **infinite** ... An Application to Platooning in **Automated** Highway Systems ...
Cited by 78 - View as HTML - Web Search

[BOOK] **Automatic** verification of finite state concurrent system using temporal logic specifications: a ...
EM Clarke, EA Emerson, AP Sistla - 1983 - ACM Press New York, NY, USA
Cited by 194 - Web Search - Library Search

Implementing Highly-Available WWW Servers based on Passive Object Replication - group of 6 »
R Baldoni, S Bonamoneta, C Marchetti - Dipartimento di Informatica e Sistemistica, Universita "La ..., 1998 - doi.ieeecs.org
... some certified name-servers (2) to automatically **modify** the DNS ... of the execution of an **infinite loop** (lines 14 ... it computes the object state **update**, the object ...
Cited by 2 - Web Search

Concept prototyping a real time switch maintenance expert system
JR Fox, GM Slawsky - Communications, 1988. ICC 88. Digital Technology-Spanning ..., 1988 - ieeexplore.ieee.org
... previously represented information and rapidly **modify** the content ... 3. VERIFICATION AND USER'S **UPDATE** TO ADVICE 4 ... Analysis module is an **infinite loop** waiting for ...
Cited by 1 - Web Search

A system to improve incorrect programs
H Wertz - Proceedings of the 4th international conference on Software ..., 1979 - portal.acm.org
... formed if it doesn't contain obvious **infinite** loops, doesn ... mete-evaluation of the **loop**-body and tries to prove ... a dual one indicating how to **modify** the program ...
Cited by 5 - Web Search

Automatic generation of C++ code from an ESCRO2 specification
PC Grabow, L Liu - Computer Software and Applications Conference, 1995. COMPSAC ..., 1995 - ieeexplore.ieee.org
... 3157/95 $04.00 © 1995 IEEE 18 **Automatic** Generation of ... ana- lyze an ESCRO2 specification using **automated** theo rem ... TRUE; boolean c_hw_green::**update**() I boolean ...
Web Search - BL Direct

Expressions for the Mean Transfer Delay of Generalized$ M$-Stage Hybrid ARQ Protocols - group of 2 »
EFC LaBerge, JM Morris - Communications, IEEE Transactions on, 2004 - ieeexplore.ieee.org
... We **modify** the SFG of Fig ... attempts that terminate in the FAIL state have **infinite** transfer delay ... this case, the transmittance of the T1-to-T1 **loop** identified as ...
Cited by 1 - Web Search - BL Direct

◀ Goooooooooo o o o o o o o o o o o o gle ▶
Result Page: **Previous** 1 2 3 4 5 6 7 8 9 **10** 11 12 13 14 15 16 17 18 19 **Next**

(endless OR infinite OR alderson) Search

Google Home - About Google - About Google Scholar

GOOGLE SCHOLAR

Published before December 1999
Terms used **infinite loop**

Found **13** of **106,921**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 13 of 13

Relevance scale

1 Infinite loops and how to create them

John R. Searle
January 1987 **ACM SIGAPL APL Quote Quad , Proceedings of the international conference on APL: APL in transition APL '87**, Volume 17 Issue 4
Publisher: ACM Press
Full text available: pdf(361.04 KB) Additional Information: full citation, abstract, references, index terms

Infinite loops are the worst enemy of any programmer in any language. They waste potentially huge amounts of time, system resources and/or money, and can destroy the credibility of the programmer amongst colleagues or customers. In APL, infinite loops can potentially occur wherever a loop is present. However, they can also occur wherever any branching statement is used, and can even occur where no branching instructions have been used. In order to avoid infinite loops, the progra ...

2 The text's the thing: Concordances to literary texts (abstract only)

Michael Preston
May 1981 **ACM SIGSOC Bulletin , Proceedings of the joint conference on Easier and more productive use of computer systems. (Part - I): Information processing in the social sciences and humanities - Volume 1981**, Volume 12-13 Issue 4-1
Publisher: ACM Press
Additional Information: full citation, abstract, index terms

The history of computer-generated concordances is already one-third of a century long. Thousands of concordances have been generated; many have been published. Most of these are useful, but there are limitations to all of them. In this presentation I discuss a number of variations on concordance-making based on specific projects being carried out at the University of Colorado.A word-form concordance can be of considerable utility. Particularly for older states of language of which our knowledge ...

3 Speech II: The procedure to construct a word predictor in a speech understanding system from a task-specific grammar defined in a CFG or a DCG

Yasuhisa Niimi, Shigeru Uzuhara, Yutaka Kobayashi
August 1986 **Proceedings of the 11th coference on Computational linguistics**
Publisher: Association for Computational Linguistics
Full text available: pdf(246.58 KB) Additional Information: full citation, abstract, references

This paper describes a method for converting a task-dependent grammar into a word

Published before December 1999
Terms used **endless loop**

Found **1** of **106,921**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 1 of 1

Relevance scale

1 Interactive control restructuring

Jeanette M. Bruno, Daniel J. Rosenkrantz
September 1994 **ACM SIGAda Ada Letters , Proceedings of the second international symposium on Environments and tools for Ada SETA2**, Volume XIV Issue SI
Publisher: ACM Press
Full text available: pdf(1.52 MB) Additional Information: full citation, abstract, index terms

An interactive algorithm for improving control flow is introduced. This algorithm has been implemented within the ENCORE re-engineering environment. The objective of the algorithm is to restructure input code so as to simplify the control flow. A key feature of the algorithm is that is permits user control during the restructuring. The algorithm also handles, in a natural manner, multiple return statements, multiple loop exits, multi-level loop exits and endless loops.

Results 1 - 1 of 1

Terms used **infinite loop edit code** Found **807** of **178,880**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 181 - 200 of 200 Result page: previous 1 2 3 4 5 6 7 8 9 **10**

Best 200 shown Relevance scale

181 The use of program profiling for software maintenance with applications to the year 2000 problem

Thomas Reps, Thomas Ball, Manuvir Das, James Larus

November 1997 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 6th European conference held jointly with the 5th ACM SIGSOFT international symposium on Foundations of software engineering ESEC '97/FSE-5**, Volume 22 Issue 6

Publisher: Springer-Verlag New York, Inc., ACM Press

Full text available: pdf(1.85 MB) Additional Information: full citation, references, citings, index terms

182 A metaobject protocol for C++

Shigeru Chiba

October 1995 **ACM SIGPLAN Notices , Proceedings of the tenth annual conference on Object-oriented programming systems, languages, and applications OOPSLA '95**, Volume 30 Issue 10

Publisher: ACM Press

Full text available: pdf(1.60 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents a metaobject protocol (*MOP*) for C++. This MOP was designed to bring the power of meta-programming to C++ programmers. It avoids penalties on runtime performance by adopting a new meta-architecture in which the metaobjects control the compilation of programs instead of being active during program execution. This allows the MOP to be used to implement libraries of efficient, transparent language extensions.

183 Revised report on the algorithmic language scheme

H. Abelson, R. K. Dybvig, C. T. Haynes, G. J. Rozas, N. I. Adams, D. P. Friedman, E. Kohlbecker, G. L. Steele, D. H. Bartley, R. Halstead, D. Oxley, G. J. Sussman, G. Brooks, C. Hanson, K. M. Pitman, M. Wand

July 1991 **ACM SIGPLAN Lisp Pointers**, Volume IV Issue 3

Publisher: ACM Press

Full text available: pdf(4.08 MB) Additional Information: full citation, abstract, citings, index terms

The report gives a defining description of the programming language Scheme. Scheme is a statically scoped and properly tail-recursive dialect of the Lisp programming language invented by Guy Lewis Steele Jr. and Gerald Jay Sussman. It was designed to have an

Nothing Found

Your search for **+infinite +loop +update +code** did not return any results.

You may want to try an Advanced Search for additional options.

Please review the Quick Tips below or for more information see the Search Tips.

Quick Tips

- Enter your search terms in lower case with a space between the terms.

  ```
  sales offices
  ```

  You can also enter a full question or concept in plain language.

  ```
  Where are the sales offices?
  ```

- Capitalize proper nouns to search for specific people, places, or products.

  ```
  John Colter, Netscape Navigator
  ```

- Enclose a phrase in double quotes to search for that exact phrase.

  ```
  "museum of natural history" "museum of modern art"
  ```

- Narrow your searches by using a + if a search term must appear on a page.

  ```
  museum +art
  ```

- Exclude pages by using a - if a search term must not appear on a page.

  ```
  museum -Paris
  ```

Combine these techniques to create a specific search query. The better your description of the information you want, the more relevant your results will be.

```
museum +"natural history" dinosaur -Chicago
```

EBSCO HOST Research Databases

Sign In to My EBSCOhost

New Search | View Folder | Preferences | Help

US PATENT AND TRADEMARK OFFICE

Basic Search | Advanced Search | Visual Search | Choose Databases | Keyword

Find: "endless loop" in Default Fields Search Cl

and in Default Fields

and in Default Fields

in Multiple Databases

Folder is

No results were found for your search query.

You may want to try your search again after following one or more of these tips:

- Check the spelling of your search terms. Correct any misspellings and re-run the search.
- To broaden your search, use the Boolean operator OR. For example, type: Siamese OR cats.

See hints for suggestions.

Refine Search | Search History / Alerts | Results

Limit your results: Limiters | Expanders

Scholarly (Peer Reviewed) Journals ☐

Published Date Month Yr: to Dec Yr: 1999

Publication

Special limiters for *Academic Search Premier*

Full Text ☐

References Available ☐

Publication Type: All, Periodical, Newspaper, Book

Document Type: All, Abstract, Article, Bibliography

Number Of Pages All

Cover Story

EBSCO HOST Research Databases

Sign In to My EBSCOhost

New Search | View Folder | Preferences | Help

US PATENT AND TRADEMARK OFFICE

Basic Search | Advanced Search | Visual Search | Choose Databases

Keyword

Find: "infinite loop" in Default Fields Search Cl

and in Default Fields

and in Default Fields

in Multiple Databases

Folder is

No results were found for your search query.

You may want to try your search again after following one or more of these tips:

- Check the spelling of your search terms. Correct any misspellings and re-run the search.
- To broaden your search, use the Boolean operator OR. For example, type: Siamese OR cats.

See hints for suggestions.

Refine Search | Search History / Alerts | Results

Limit your results:

Limiters | Expanders

Scholarly (Peer Reviewed) Journals ☐

Published Date Month Yr: to Dec Yr: 1999

Publication

Special limiters for *Academic Search Premier*

Full Text ☐

References Available ☐

Publication Type: All, Periodical, Newspaper, Book

Document Type: All, Abstract, Article, Bibliography

Number Of Pages All

Cover Story

Results for "(('endless loop'<in>ab)) <and> (pyr >= 1951 <and> pyr <= 1999)" e-mail
Your search matched **3** of **1365662** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

» Search Options

View Session History

New Search

Modify Search

(('endless loop'<in>ab)) <and> (pyr >= 1951 <and> pyr <= 1999) Search

☐ Check to search only within this results set

Display Format: ○ Citation ◉ Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All

☐ **1. Noise from anhysteretic remanence in magnetic tapes**
Lindholm, D.;
Magnetics, IEEE Transactions on
Volume 7, Issue 2, Jun 1971 Page(s):312 - 315
Summary: A technique is described for producing anhysteretic remanent magi in situ on an endless loop tape recorder. A comparison of noise from ARM with from room temperature isothermal remanent magnetization (IRM) is presente..
AbstractPlus | Full Text: PDF(424 KB) **IEEE JNL**
Rights and Permissions

☐ **2. Real-time performance improvements for distributed databases on a wid**
Reddan, C.; Tabak, D.;
Real Time, 1990. Proceedings., Euromicro '90 Workshop on
6-8 June 1990 Page(s):216 - 223
Digital Object Identifier 10.1109/EMWRT.1990.128254
Summary: An effort to improve the performance of distributed processing appl high-speed wide area networks is described. The primary limiting factor in this be latency, the innate propagation delay connected with sending signal.....
AbstractPlus | Full Text: PDF(516 KB) **IEEE CNF**
Rights and Permissions

☐ **3. Wave follower instrumentation platform redesign and test**
Harris, D.B.; DeCicco, D.J.;
OCEANS '93. 'Engineering in Harmony with Ocean'. Proceedings
18-21 Oct. 1993 Page(s):I439 - I443 vol.1
Digital Object Identifier 10.1109/OCEANS.1993.326052
Summary: The Office of Naval Research (ONR) sponsored the design and fat wave follower system. The wave follower is a mechanical device designed to p above and below the sea surface. Instruments are mounted on a platform susp
AbstractPlus | Full Text: PDF(324 KB) **IEEE CNF**
Rights and Permissions

Results for "(('infinite loop'<in>ab)) <and> (pyr >= 1951 <and> pyr <= 1999)"

e-mail

Your search matched **5** of **1365662** documents.

A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

» Search Options

View Session History

New Search

Modify Search

(('infinite loop'<in>ab)) <and> (pyr >= 1951 <and> pyr <= 1999) Search

Check to search only within this results set

Display Format: Citation Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All

1. **Dynamics of one-cycle controlled Cuk converters**
   Smedley, K.M.; Cuk, S.;
   Power Electronics, IEEE Transactions on
   Volume 10, Issue 6, Nov. 1995 Page(s):634 - 639
   Digital Object Identifier 10.1109/63.471282
   Summary: One-cycle control is a nonlinear control method. The flow-graph mc is employed to study the large-signal and small-signal dynamic behavior of one switching converters. Systematic design method for one-cycle control sy.....
   AbstractPlus | Full Text: PDF(516 KB) **IEEE JNL**
   Rights and Permissions

2. **An always-convergent method for finding the DC operating points of non**
   Szabo, R.;
   Circuits and Systems, 1998. IEEE APCCAS 1998. The 1998 IEEE Asia-Pacific
   24-27 Nov. 1998 Page(s):355 - 358
   Digital Object Identifier 10.1109/APCCAS.1998.743773
   Summary: The Newton-Raphson algorithm is the most widely used nonlinear s yet it has the serious drawback that, depending on its initial point, it can easily loop or diverge to infinity. This paper presents a simple yet pre.....
   AbstractPlus | Full Text: PDF(224 KB) **IEEE CNF**
   Rights and Permissions

3. **A fault tolerant distributed parallel processing system on LAN workstatio**
   Haghighat, A.T.; Faez, K.;
   Information, Communications and Signal Processing, 1997. ICICS., Proceedin International Conference on
   Volume 3, 9-12 Sept. 1997 Page(s):1432 - 1435 vol.3
   Digital Object Identifier 10.1109/ICICS.1997.652228
   Summary: Workstations of a computer network can be used to implement a p system. In this method the existing network equipment can be used without an cost for parallel processing. The aim of this paper is the development of a.....
   AbstractPlus | Full Text: PDF(356 KB) **IEEE CNF**
   Rights and Permissions

4. **Pulling motion based tactile sensing for concave surface**
   Kaneko, M.; Higashimori, M.; Tsuji, T.;
   Robotics and Automation, 1997. Proceedings., 1997 IEEE International Confer
   Volume 3, 20-25 April 1997 Page(s):2477 - 2484 vol.3
   Digital Object Identifier 10.1109/ROBOT.1997.619333

This paper examines a family of program test data selection criteria derived from data f techniques similar to those used in compiler optimization. It is argued that currently us selection criteria which examine only the control flow of a program are ...

Result # 48 Relevance:

Control System for Self Calibrating Digital Converter Offsets in an Ana

1996-06-01 IPCOM000117796D English

Analog circuits have offset voltages that can vary with usage conditions and subtract fr dynamic range of the circuits. This disclosure describes a system that calibrates out the means of a 1 bit Analog to Digital Converter (ADC), a logic control ...

Result # 49 Relevance:

Automatic Multiple Concurrent Package Skew Minimization Algorithm

1997-07-01 IPCOM000118798D English

Disclosed is an algorithm which automatically minimizes skew across multiple packages selected set of nets. Skew is the difference in time between the earliest and latest signa within a group or bundle of nets.

Displaying page 5 of 5 << FIRST | < BACK | NEXT > | LAST >>

Search query: "endless loop"

Published Before: 12-15-1999 (Original publication date)

New search | Modify this search | Search within current results

Displaying records #41 through 41 out of 41

Result # 41 Relevance:

MTS The Michigan Terminal System Volume 7: PL/I in MTS

1985-09-01 IPCOM000128754D English

The software developed by the Computing Center staff for the operation of the high-spe computer can be described as a multiprogramming supervisor that handles a number o reentrant programs. Among them is a large subsystem, called MTS (Michigan Terminal

Displaying page 5 of 5 << FIRST | < BACK | NEXT > | LAST >>

Search query: "infinite loop"

Published Before: 12-15-1999 (Original publication date)

New search | Modify this search | Search within current results

ProQuest®

Results – powered by ProQuest® Smart Search

Suggested Topics About < Previous | Next >

Antennas

5 documents found for: *("endless loop") AND PDN(<12/15/1999)* » Refine Search | Set Up Alert

All sources | Trade Publications | Dissertations

☐ Mark all 0 marked items: Email / Cite / Export | Show only full text | Sort results by: Most recent first

1. **Cardinal IDC 100i**
Anonymous. **Computer Reseller News.** May 19, 1997. p. 152 (1 page)
Full text | Page Image - PDF | Abstract

2. **Killing runaways**
Cummings, Joanne. **Computerworld.** Framingham: Oct 4, 1993. Vol. 27, Iss. 40; p. 113 (1 page)
Full text | Page Image - PDF | Abstract

3. **Implicaciones medico-quirurgicas de la conduccion ventriculo-auricular con aplicacion al control de las taquicardias de asa cerrada**
by *Gascon Lopez, Damian*, M.Med., **Universidad de Sevilla (Spain)**, 1990, 260 pages; AAT C318812
Abstract

4. **Statische Analyse von PROLOG-Programmen**
by *Reichl, Franz*, Dr.Tech., **Technische Universitaet Wien (Austria)**, 1988, 167 pages; AAT C105817
Abstract

5. **The Impact and Non-Impact of Printers**
Young, Robert P.. **Small Systems World.** Dec 1979. Vol. 7, Iss. 6; p. 24
Abstract

1-5 of 5

Want to be notified of new results for this search? Set Up Alert Results per page: 30

Did you find what you're looking for? If not, refine your search below or try these suggestions.

Suggested Topics About < Previous | Next >

Antennas

PROQUEST

ProQuest®

Results – powered by ProQuest® Smart Search

Suggested Topics About < Previous | Next >

Antennas

28 documents found for: *("infinite loop") AND PDN(<12/15/1999)* » Refine Search | Set Up Alert

All sources Scholarly Journals Trade Publications Dissertations

☐ Mark all 0 marked items: Email / Cite / Export Show only full text Sort results by: Most r

1. **Getting to the core**
Dean Takahashi. **Upside (U.S. ed.).** Foster City: Jul 1999. Vol. 11, Iss. 7; p. 160 (2 pages)
Full text Page Image - PDF Abstract

2. **Bug++ of the month**
Ron Burk. **Windows Developer's Journal.** Jun 1999. Vol. 10, Iss. 6; p. 65 (2 pages)
Abstract

3. **Biting the apple that feeds us**
Hinman, Roderick T. **IEEE Spectrum.** New York: May 1999. Vol. 36, Iss. 5; p. 16 (2 pages)
Abstract

4. **Exposing the deep, dark secrets of TCP/IP**
Brad Turner. **Network World.** Framingham: Feb 1, 1999. Vol. 16, Iss. 5; p. 33 (1 page)
Text+Graphics Page Image - PDF Abstract

5. **Spanning the LAN; The author of spanning tree also has turned her hand to other literary pursuits**
Radia Perlman. **Data Communications.** New York: October 21, 1997. Vol. 26, Iss. 14; p. 68
Abstract

6. **Decomposition of tautologies into regular formulas and strong completeness of connection-graph**
W Bibel, E Eder. **Association for Computing Machinery. Journal of the Association for Computing M**
York: Mar 1997. Vol. 44, Iss. 2; p. 320 (25 pages)
Text+Graphics Page Image - PDF Abstract

7. **Coherence for iterated monoidal categories and homological obstructions to delooping**
by *Balteanu, Comel*, Ph.D., **The Ohio State University**, 1997, 81 pages; AAT 9801638
Abstract 24 Page Preview Page Image - PDF Order a

8. **NT weakness could cause service outages**
Trott, Bob. **InfoWorld.** San Mateo: Dec 23/30, 1996. Vol. 18, Iss. 52/53; p. 16 (1 page)

RESEARCH DISCLOSURE

Home Overview Publi

Search RD Database

Database last u

Log in

Search
- Quick Search
- Advanced Search
- Numeric Search
- Recent Search
- Last Result Set

Recent Disclosures

Search parameters information



Quick Search | Advanced Search | Numeric Search | Recent Searches

Results for search query

All = "endless loop"
From Jan 1960 To Dec 1999

8 Disclosures found Refine search Display 50 results

Page: [1] Dow

RD ID	Disclosure title
403005	Watchdog to guard correct micro processor operation
173015	Improvements relating to sanding belt guidance
267014	Method of forming a seamless tractor drive belt
227040	'Dymetrol' elastomeric tape drives automotive energy transfer systems
207016	Improvements in spiral seams for papermachine felts and fabrics
134022	Video recording system
181029	A method for determining exposure times for a color printer
155035	Improved light control and display devices

Page: [1] Dow

RESEARCH DISCLOSURE

Home Overview Publi

Search RD Database

Database last u

Log in

Search
- Quick Search
- Advanced Search
- Numeric Search
- Recent Search
- Last Result Set

Recent Disclosures

Search parameters information

Quick Search | Advanced Search | Numeric Search | Recent Searches

Results for search query

All = "infinite loop"
From Jan 1960 To Dec 1999

0 Disclosures found Refine search

SCIENCE DIRECT Register or Login: user name Password: Go Athens/Institution Log

Home | Search | Journals | Books | Abstract Databases | My Profile | Alerts | Help

Quick Search: within All Full-text Sources Go Search Tips

No results were found
Click the search tips link on the search form below for additional information.

All Sources | Journals | Books | Reference Works | Abstract Databases | Scirus

Basic | Advanced

Term(s): "endless loop" within: Abstract

AND within: Abstract, Title, Keywords

Sources: Journals Book Series Handbooks Reference Works Abstract Databases

Subject: select one or more:

- All Sciences -
Agricultural and Biological Sciences
Arts and Humanities
Biochemistry, Genetics and Molecular Biology

Hold down the Ctrl key (or ⌘ key) to select multiple entries.

Dates: 1947 to: 1999 All Years

Search | Clear | Recall Search | Search Tips

Search History - Turn On

Search for articles from our full-text collection and abstracts database using this search form. Click the **Help** button for step-by-step instructions on conducting a search using this form. Consult the Search Tips for information about the use of connectors, wildcards, and other search options which can improve the precision of your search.

Home | Search | Journals | Books | Abstract Databases | My Profile | Alerts | Help

SCIENCE DIRECT

results **1 - 15**

15 Articles Found

pub-date > 1946 and pub-date < 2000 and ABSTRACT("infinite loop")

1. ☐ **THE *K*-THEORY LOCALIZATIONS AND v_1-PERIODIC HOMOTOPY GROUPS OF *H*-SPACES** • ARTICLE
Topology, Volume 38, Issue 6, November 1999, Pages 1239-1264
A. K. Bousfield
Abstract | PDF (258 K)

2. ☐ **Appearance of hierarchical structure in hyper-dilation model: model of generalized measurement process** • ARTICLE
Applied Mathematics and Computation, Volume 104, Issues 2-3, September 1999, Pages 153-178
Shin'ichi Toyoda and Pegio-yukio Gunji
Abstract | Abstract + References | PDF (952 K)

3. ☐ **Infinite loop spaces with odd torsion-free homology** • ARTICLE
Topology, Volume 37, Issue 2, March 1998, Pages 329-338
Michael Slack
Abstract | Abstract + References | PDF (693 K)

4. ☐ **A rewrite mechanism for logic programs with negation** • ARTICLE
Theoretical Computer Science, Volume 192, Issue 1, 10 February 1998, Pages 77-106
Siva Anantharaman and Gilles Richard
Abstract | Abstract + References | PDF (2271 K)

5. ☐ **Constraint-based design for 3D shapes** • ARTICLE
Artificial Intelligence, Volume 91, Issue 1, March 1997, Pages 51-69
Shuichi Shimizu and Masayuki Numao
Abstract | Abstract + References | PDF (1223 K)

6. ☐ **Solving a Unification Problem under Constrained Substitutions Using Tree Automata** • ARTICLE
Journal of Symbolic Computation, Volume 23, Issue 1, January 1997, Pages 79-117
YUICHI KAJI, TORU FUJIWARA and TADAO KASAMI
Abstract | PDF (875 K)

CRCnetBASE

Search GO

Advanced Search Help

Online Products

Contact Us

Lib Corner

News

FAQs

Home

Search Results

44 results for **"endless loop"**

- ⊞ **InfoSECURITYnetBASE** 9 results
- ⊞ **BUSINESSnetBASE** 8 results
- ⊞ **ENGnetBASE** 7 results
- ⊟ **ElectricalEngineeringnetBASE**
  - ⊞ **The Telecommunications Illustrated Dictionary, Secc Edition** 2 results
  - ⊞ **Domain-Specific Processors: Systems, Architectures, Modeling, and Simulation** 1 results
  - ⊞ **Handbook of Ad hoc Wireless Networks (The)** 1 result
  - ⊟ ***LabVIEW: Advanced Programming Techinques***
    - Chapter 3: State Machines (4 Hits)
- ⊞ **PublicADMINISTRATIONnetBASE** 4 results
- ⊞ **ProjectMANAGEMENTnetBASE** 4 results
- ⊞ **TELECOMMUNICATIONSnetBASE** 3 results
- ⊞ **POLYMERSnetBASE** 1 results
- ⊞ **FORENSICnetBASE/ LAWENFORCEMENTnetBASE**
- ⊞ **CHEMLIBnetBASE** 1 results
- ⊞ **NEUROSCIENCEnetBASE** 1 results

CRCnetBASE

Advanced Search InterJournal Manuscripts

This form allows you to search the accepted and publicly available submitted manuscripts of InterJournal

Leave the 'phrase' field blank to list all manuscripts.

	Word, Phrase		Field
Search for	"endless loop"	in	Abstract
And		in	Title
And		in	Title
And		in	Title

Submitted since: (*Format* = yymmdd)
Submitted before: 991215 (*Format* = yymmdd)

Start Search Basic Search

Advanced Search InterJournal Manuscripts

This form allows you to search the accepted and publicly available submitted manuscripts of InterJournal

Leave the 'phrase' field blank to list all manuscripts.

	Word, Phrase		Field
Search for	"infinite loop"	in	Abstract
And		in	Title
And		in	Title
And		in	Title

Submitted since: (*Format* = yymmdd)
Submitted before: 991215 (*Format* = yymmdd)

Start Search

Basic Search

Search results

No matches were found for 'endless and loop and '

Check the spelling of the search word(s) you used. If the spelling is correct and you only used one word, try using one or more similar search words with "**Any**."

If the spelling is correct and you used more than one word with "**Any**," try using one or more similar search words with "**Any**."

If the spelling is correct and you used more than one word with "**All**", try using one or more of the same words with "**Any**."

Match: All Format: Long
Refine search: Search

ht://Dig 3.0

Search results for 'infinite and loop and '

Match: All Format: Long
Refine search: Search

Documents 1 - 10 of 10 matches. More ☆'s indicate a better match.

Untitled Document☆☆☆☆

. . . with J.T. Oden (University of Texas at Austin). DR VLADIMIR MARKOVIC of the University of Warwick is awarded a Whitehead prize for his work on **infinite**-dimensional Teichmüller spaces. Markovic, with colleagues, has resolved a number of outstanding open questions, starting in his thesis (1998) with a . . .
http://www.lms.ac.uk/newsletter/328/328_01.html , 51258 bytes

Proceedings of the London Mathematical Society: cumulative author index☆☆☆

. . . Bernd The Loewy series of the Steinberg-PIM of finite general linear groups, 92 (2006) 62-98 Adeleke, S. A. and Macpherson, Dugald Classification of **infinite** primitive Jordan permutation groups, 72 (1996) 63-123 Adler, Allan The Mathieu group M 11 and the modular curve X(11), 74 (1997) 1-28 Agboola . . .
http://www.lms.ac.uk/publications/proceedings/pcumind.html , 179609 bytes

LMS Proceedings Cumulative Index, Volumes 72-89☆☆☆

. . . Freitas, Pedro On the invariant spectrum of S 1-invariant metrics on S 2, 84 (2002) 213-230 Adeleke, S. A. and Macpherson, Dugald Classification of **infinite** primitive Jordan permutation groups, 72 (1996) 63-123 Adler, Allan The Mathieu group M 11 and the modular curve X(11), 74 (1997) 1-28 Agler, J . . .
http://www.lms.ac.uk/publications/proceedings/pcumind-oldstyle.html , 141880 bytes

Newsletter Item☆☆

. . . analysis. Among these, both for significance and originality, one could cite her work on the equivalence of different notions of null sets in **infinite** dimensional spaces, the construction of a Lipschitz quotient between space of different - finite - dimensions which does not satisfy Gorelik's principle . . .
http://www.lms.ac.uk/newsletter/0207/announcements.html , 17648 bytes

Untitled Document☆☆

. . . homomorphisms'. If H is a graph, then the problem of deciding whether an arbitrary graph has an H-colouring is NP-complete unless H has either a **loop** or a bipartite component (in which case the problem is trivially in P). Exact counting is #P-complete unless every component is either a complete looped . . .
http://www.lms.ac.uk/newsletter/343/343_02.html , 11274 bytes

Bulletin of the London Mathematical Society Forthcoming Papers☆☆

JOURNAL
ABOUT e-SJIS
SUBSCRIPTION
EDITORIAL POLICY
SEARCH CONTENT
FORTHCOMING PAPERS
SJIS ARCHIVE

FOR AUTHORS
EDITORIAL BOARD
ADVISORY BOARD
SUBMISSION OF PAPERS
PROPOSALS - SPECIAL ISSUE

SERVICES
ON-LINE SUBMISSION
E-ALERTS
PRICES
CONTACT

RESOURCES
IRIS ASSOCIATION

SEARCH CONTENT

Welcome to the search page of SJIS. Use the form below to search for key words and past issues of the journal. Search is at the moment restricted to only one author at a ti title or body text phrase at the time.

Please select search attribute and enter appropriate search string below

Search attribute: Body text

Search string: "endless loop"

Search! Reset

TOP

JOURNAL
ABOUT e-SJIS
SUBSCRIPTION
EDITORIAL POLICY
SEARCH CONTENT
FORTHCOMING PAPERS
SJIS ARCHIVE

FOR AUTHORS
EDITORIAL BOARD
ADVISORY BOARD
SUBMISSION OF PAPERS
PROPOSALS - SPECIAL ISSUE

SERVICES
ON-LINE SUBMISSION
E-ALERTS
PRICES
CONTACT

RESOURCES
IRIS ASSOCIATION

SEARCH CONTENT

Welcome to the search page of SJIS. Use the form below to search for key words and past issues of the journal. Search is at the moment restricted to only one author at a ti title or body text phrase at the time.

Please select search attribute and enter appropriate search string below

Search attribute: Body text

Search string: "infinite loop"

Search! Reset

TOP